

## Mask Set Errata

9S12A128MSE1  
3/2002

Mask Set Errata  
MC9S12A128 8-Bit  
Microcontroller Unit



---

## Introduction

This errata provides information applicable to the 0L85D MCU mask set of the 9S12A128.

---

## MCU Device Mask Set Identification

The mask set is identified by a four-character code consisting of a letter, two numerical digits, and a letter, for example F74A. Slight variations to the mask set identification code may result in an optional numerical digit preceding the standard four-character code, for example 0F74A.

---

## MCU Device Date Codes

Device markings indicate the week of manufacture and the mask set used. The data is coded as four numerical digits where the first two digits indicate the year and the last two digits indicate the work week. The date code "9115" would indicate the 15th week of the year 1991.

---

## MCU Device Part Number Prefixes

Some MCU samples and devices are marked with an SC, PC, ZC or XC prefix. An SC, PC or ZC prefix denotes special/custom device. An XC prefix denotes device is tested but is not fully characterized or qualified over the full range of normal manufacturing process variations. After full characterization and qualification, devices will be marked with the MC prefix.

---

## Errata System Tracking Numbers

MUCTS00xxx is the tracking number for device errata. It can be used with the mask set and date code to identify a specific errata to a Motorola representative.

### SPI Locks if Disabled During Message Transmission

In master mode during a transmission SPI locks if SPE bit is cleared. After re-enabling, writing to SPIDR does not result in message transmission.

**Workaround**                      Disable the SPI module only if transmission queue is empty (SPTEF=1) and transfer is complete (SPIF=1).

### Clock Monitor Frequency Lower than Specified

The clock monitor failure assert frequency is  $f_{CMFA} = (\text{max:100kHz, typ:50kHz, min:25kHz})$  and not as the specified  $f_{CMFA} = (\text{max:200kHz, typ:100kHz, min:50kHz})$ .

**Workaround**                      None

### Self Clock Frequency Too High

The self clock mode frequency can exceed the maximum specified value. In this case the clock quality check will always fail for EXTAL frequencies below 800kHz. This affects two additional spec values.

1. The clock quality check may fail for EXTAL frequencies below 800kHz. Therefore the crystal oscillator frequency range minimum value is 1.0MHz.
2. The PLL may not lock to the minimum specified PLL frequency of 8.0MHz. Therefore the VCO locking range minimum value is 12.0MHz, which corresponds to a 6.0MHz bus frequency.

**Workaround**

1. Instead of 500kHz use a 1MHz quartz, resonator or oscillator.
2. Only synthesize PLL frequencies from 12MHz to the maximum specified value. 12MHz PLL frequency corresponds to 6MHz bus frequency.

### PWM Channel Early Start after Leaving Emergency Shutdown Mode

When recovering from the emergency shutdown mode by disasserting the active level on the PWM emergency shutdown input pin and subsequently asserting the PWMRSTRT bit, the enabled PWM channels do not hold the shutdown output level (PWMLVL) until the corresponding counter passes zero. This may result in a pulse of undefined length on enabled PWM channels.

**Workaround**                      None

---

### Breakpoint Module: Potential Extraneous Data Match

When using the breakpoint in full mode with the read/write match function disabled, there is a chance of a false match. Internally there is a separate read data bus and write data bus. When in full mode with the read/write match function is disabled, both buses are always compared to the contents of data match register. The circuit should only match the active bus on any particular bus cycle. The false match can occur if the address matches on a read cycle and matching data is on the write data bus or the address matches on a write cycle and the matching data is on the read data bus.

#### Workaround

When using full mode, always enable the read/write match and select the appropriate state of read/write. This will avoid false matches.

---

### ATD: Incorrect Offset of Transfer Curve for 8-Bit Resolution

8-bit mode transfer characteristic shows incorrect result of 17.5mV for first transition. This is because independently of 8- or 10-bit resolution an offset of 2.5mV is used (which is only correct for 10-bit resolution).

#### Workaround

None

---

### SPI Can Receive Incorrect Data in Slave Mode

An SPI configured for slave mode operation can receive incorrect data. If there are clock edges on SCK while SPE=0, and then SPE is set to one, the received data will be incorrect. In CPHA=1 mode the SPI will continue to receive incorrect data as long as SPE=1.

#### Workaround

Depending on the current mode on SPI, the following bits have to be configured while disabling the SPI:

- Set CPHA=1 and make sure that the CPOL-bit is clear every time the SPI is disabled from slave mode.
- Clear CPHA and make sure that CPOL is clear as well every time the SPI is disabled from master mode.

## SPIF Flag is Set Wrong in Slave Mode after SPI Re-enabling

The SPIF-flag is set wrong under the following conditions:

- Receiving a byte with the SPI while configured as slave, CPHA=1 and CPOL=1
- Clearing SPIF in SPISR normally (read SPISR followed by a read of SPIDR)
- Disabling the SPI through clearing SPICR1
- Re-enabling the SPI (with CPHA=1 and CPOL=1)

Three bus cycles after this sequence SPIF in SPISR is set, which is not correct.

### Workaround

A sequence to clear the wrong SPIF is:

If no other IRQs enabled:

- Insert a short delay (min. 3 bus cycles) after re-enabling the SPI and clear the SPIF by reading the SPISR followed by SPIDR after re-enabling the SPI.

If other IRQs enabled (excluding X-IRQ):

- Before re-enabling the SPI, disable all IRQs through a set of bit CCR[4] (IRQ mask) using the instruction SEI
- Re-enable SPI, insert a short delay (min. 3 bus cycles) after SPI re-enabling
- Clear SPIF by a read access to the SPISR followed by SPIDR.
- Re-enable all IRQs through clearing bit CCR[4] using the instruction CLI

In the case where a hardware interrupt is used, it is possible to lose an SPI IRQ.

## SPI Locks if Re-enabled as Master

The SPI locks if it is disabled in master mode with CPHA=1 in SPICR1 and re-enabled in master mode with CPHA=1.

### Workaround

Make sure that CHPA is not set when SPI is disabled after a transmission in master mode.

---

## Non-Multiplexed Addresses on PK Change Before End of Cycle

In expanded modes with the EMK emulate port k bit set and the EXSTR[1:0] external access stretch bits 1 & 0 set to 01, 10 or 11 the non-multiplexed addresses on PK[5:0] change during E clock high phase at t4.

### Workaround

If the external access is stretched (EXSTR[1:0] set to 01, 10 or 11) off chip address latches should be used to register the non-multiplexed addresses on PK[5:0].

---

## SCI Interrupt Asserts Only if an Odd Number of Interrupts Active

The interrupt of the SCI is only asserted if an odd number of interrupts is active (i.e., flags set and enabled). Example: If an Transmit data register empty and an receive ready interrupt are active at the same time the interrupt request to the CPU is not asserted. This can lead to missing interrupts or spurious interrupts i.e. the request gets deasserted before the CPU fetches the interrupt vector. Those spurious interrupts will execute on the SWI vector. The interrupt flag setting is always correct.

### Workaround

There is no general workaround. Some typical cases will be described. The problem is reduced by fast interrupt response times and slow baud rates.

1. Single wire interfaces as used in the automotive industry Since the transmit and receive process are linked to each other a mix of interrupt handling and polling is possible. Here several scenarios are possible to come to a stable operation but each might require a rewriting of the lower level drivers. The easiest one is to use only receive interrupt and fill the first two bytes of a message into the transmit queue by polling. With each byte received the transmit data buffer is empty and a new byte can be queued.
2. Full duplex operation If the SCI interrupt is not asserted while e.g. a transmit or receive interrupt are pending the received byte will cause an overflow error (3rd interrupt) and this will then assert the interrupt. A software can detect the overrun error and request a re-transmission of the last message frame. One byte is lost here. Polling flag bits at a frequency of one byte time. (E.g. 1ms for 9600 bd) Toggling one of the enable flags at the frequency of one byte.

---

## PE7 (XCLKS) is Not Pulled Up Internally when Reset in Emulation Mode

When the MCU is reset in emulation expanded wide or narrow mode, PE7 (XCLKS) is not pulled up internally. In that case, PE7 needs to be driven by an external device.

### Workaround

When MCU is reset in expanded emulation wide or narrow mode, PE7 (XCLKS) needs to be driven by an external device during RESET low.

### SPIF Flag is Set Wrong in Slave Mode

Assumption:

SPI is enabled in slave mode of operation, CPHA bit is set, all other bits are in their reset state, SS and SCK pin are driven low

Action:

CPHA bit is cleared

Result:

Three Bus Clock cycles after clearing CPHA bit, SPIF flag is set

#### Workaround

Change of CPHA bit should only occur while SPI is disabled (SPE bit cleared).

### SPIF Flag is Set Wrong — SPI Locks in Master Mode

The SPIF flag is set wrongly after the following sequence:

1. SPI receives a byte in slave mode  
SPIF flag is cleared normally (read SPISR, read SPIDR)
2. SPI is disabled through the following sequence  
(given by workaround for MUCts00488 and MUCts00492)  
SPICR1 is set to 0x04h
3. SPI is re-enabled as master through the following sequence  
(given by workaround for MUCts00488 and MUCts00492)  
SPICR1 is set to 0x08h, SPICR1 is set to 0x54h  
Three bus cycles after SPICR1 was set to 0x08h, the SPIF flag is set wrongly and also the master locks

#### Workaround

1. SPI receives a byte in slave mode  
SPIF flag is cleared normally (read SPISR, read SPIDR)
2. Disable SPI through the following sequence  
(given by workaround for MUCts00488 and MUCts00492)  
Set SPICR1 to 0x04h
3. Re-enable SPI as master through the following sequence  
(given by workaround for MUCts00488 and MUCts00492)  
Set SPICR1 to 0x0Ch (actual workaround for MUCts00560)  
Set SPICR1 to 0x08h  
Set SPICR1 to 0x54h

---

### Missing External ECLK During Reset Vector Fetch

The reset conditions of the ECLK control logic in the MEBI inhibit the generation of 1 ECLK pulse during the reset vector fetch. This can prevent the external logic from latching the reset vector address.

**Workaround**                      None

---

### SPIDR is Writable though the SPTEF Flag is Cleared

Data can be placed into the SPI Data Register (SPIDR) though the SPTEF flag is cleared. The SPTEF flag indicates, if the transmit buffer is empty (SPTEF=1) or full (SPTEF=0). Data can be placed into the SPI Data Register by reading SPISR with SPTEF=1 followed by a write to the SPI Data Register. If SPTEF=0, a write to the SPI Data Register should be ignored, according to the SPI specification. This is not true for the current implementation, where data can be placed into the SPI Data Register though SPTEF=0.

**Workaround**                      Do not write to the SPI Data Register until you have read SPISR with SPTEF=1.

---

### Erase Verify Impact on Subsequent Erase Operations

If the Erase Verify (\$05) command is issued on an array that is not erased as indicated by the FSTAT/ESTAT BLANK bit not being set upon command completion, the execution of the Sector Erase (\$40) or Mass Erase (\$41) command will not properly erase the intended region. The Program (\$20) command will execute properly.

**Workaround**                      If the Erase Verify (\$05) command is issued on an array that is not erased, subsequent Sector Erase (\$40) or Mass Erase (\$41) commands must be issued twice before the intended region is properly erased.

**HOW TO REACH US:****USA/EUROPE/LOCATIONS NOT LISTED:**

Motorola Literature Distribution;  
P.O. Box 5405, Denver, Colorado 80217  
1-303-675-2140 or 1-800-441-2447

**JAPAN:**

Motorola Japan Ltd.; SPS, Technical Information Center,  
3-20-1, Minami-Azabu Minato-ku, Tokyo 106-8573 Japan  
81-3-3440-3569

**ASIA/PACIFIC:**

Motorola Semiconductors H.K. Ltd.;  
Silicon Harbour Centre, 2 Dai King Street,  
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong  
852-26668334

**TECHNICAL INFORMATION CENTER:**

1-800-521-6274

**HOME PAGE:**

<http://www.motorola.com/semiconductors>

Information in this document is provided solely to enable system and software implementers to use Motorola products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.



Motorola and the Stylized M Logo are registered in the U.S. Patent and Trademark Office. digital dna is a trademark of Motorola, Inc. All other product or service names are the property of their respective owners. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

© Motorola, Inc. 2002

Additional mask set erratas can be found on the World Wide Web at <http://motorola.com/semiconductors>.